JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Exploring the Synergy of NTRU and BZIP2 for Enhanced Data Security and Efficiency

Abhishek Jain, Guide: Shayam S

Student, Jain (Deemed-to-Be) University, Bangalore.
, Assistant Professor, Jain (Deemed-to-Be) University, Bangalore.

Abstract:

In the digital age, where data transmission volume and cyber threat sophistication are constantly rising, the relationship between efficiency and data security is an essential field of research. The combination of BZIP2 compression, a highly effective data reduction method, and the quantum-resistant algorithm NTRU encryption is investigated in this paper. The goal is to create a system that optimizes data transmission and storage while safeguarding it against sophisticated cryptographic attacks. NTRU encryption is a strong option for future-proof data security because it is based on the difficulty of lattice problems, which are difficult for quantum computers to solve. Complementing NTRU, BZIP2 compression minimizes the data size, improving transmission speed and requiring less storage space thanks to its lossless data reduction. Impenetrable security and increased efficiency are the two benefits that this technology synergy promises.

The present study employs a methodology that entails a comprehensive examination of both NTRU and BZIP2, succeeded by the development and execution of a framework that integrates them. Thorough testing is used to assess the integrated system's performance, with an emphasis on operational effectiveness, encryption strength, and compression ratio. The outcomes show that the combined strategy significantly reduces the amount of data while maintaining data integrity and confidentiality. This dual advantage is particularly relevant in scenarios where large volumes of sensitive data are transmitted or stored. According to the study's findings, NTRU encryption combined with BZIP2 compression provides a workable answer to the problems associated with data security and efficiency in the contemporary digital environment.

Index Terms — Cryptography, NTRU encryption, Bzip2 algorithm.

Introduction:

Safeguarding data transmission is a top priority in the ever-changing world of digital communication. The overall objective of this study is to improve data security and operational effectiveness by investigating possible synergies between NTRU encryption and Bzip2 compression. NTRU, which is well-known for its strong security features and ability to work with small key sizes, shows promise as a communications security solution, especially in situations with limited resources. Simultaneously, Bzip2, a well-known compression method, is excellent at reducing the amount of data that has to be sent, which improves operational effectiveness.

The purpose of this work is to investigate how NTRU and Bzip2 may work together to maximize computational resource consumption and data security by using their unique advantages. The ability of NTRU to provide strong security with small key sizes fits in well with Bzip2's ability to compress data efficiently. It

is expected that the combination of these technologies would result in a complete solution that addresses the efficiency and secrecy issues associated with digital communications.

The main goals include a comprehensive assessment of the performance of the NTRU + Bzip2 method. In accordance with strict guidelines provided by the Institute of Electrical and Electronics Engineers (IEEE), this evaluation will include a wide range of factors, such as data transmission efficiency, computational complexity, and security metrics. The research technique comprises a methodical approach that includes empirical trials, theoretical analysis, and comparative performance assessments against industry-standard encryption and compression algorithms.

Through examining the interrelationships between NTRU encryption and Bzip2 compression, this study seeks to provide insightful contributions to the fields of data compression and cryptography. The study's results should contribute to our knowledge of each of these technologies separately as well as show how they might operate together to provide a solid, safe, and effective foundation for digital communication networks.

As we begin this thorough investigation, the next sections will examine the fundamental ideas behind Bzip2 compression and NTRU encryption, clarifying each technology's unique advantages and functions. The complexity of the suggested algorithm will then be revealed in later chapters, along with thorough analysis, experimental findings, and comparisons with industry standards. We hope that this thorough analysis will provide a comprehensive and authoritative contribution to the disciplines of data compression and cryptography, paving the way for future developments in safe and effective digital communication systems.

Problem Statement:

Combining NTRU encryption with BZIP2 compression offers a unique way to maximize transmission efficiency while protecting data from any dangers posed by quantum computing. The digital world is always changing, and new technological developments are always challenging security assumptions. Because of the rapid advancements in computing power, conventional encryption methods, which have long been the foundation of secure communication, are in danger of becoming outdated. There is now a race on to create cryptographic techniques that can resist the attack of these formidable machines due to the arrival of quantum computers. There has never been a more pressing demand for encryption methods that are both efficient and safe.

The efficiency of current cryptographic systems is sometimes lacking, especially when deploying them in contexts with constrained processing resources. The limitations of low-power, embedded, and Internet of Things devices, together with the performance needs of contemporary systems, necessitate a reassessment of current approaches. The ongoing pursuit of secure lightweight encryption is evidence of the evolving needs of the digital era.The goal of the research article at hand is to investigate how NTRU encryption and BZIP2 compression work together to improve data handling efficiency and security. With its small key sizes and quantum-resistant characteristics, NTRU encryption presents a viable way out of the security conundrum. In the meanwhile, the necessity for effective data transmission and storage is efficiently met by BZIP2 compression's capacity to minimize data size.

The combination of these two technologies has the potential to completely change how we handle data management and security. Examining the computational overhead, the strength of the encryption, the efficiency of the compression, and the system's overall performance, the study seeks to analyze the feasibility of this integration. Through examining these facets, the research hopes to provide insight into the feasibility of an integrated strategy and its suitability for use in a range of industries. It is hoped that the results of this study will significantly advance the area of lightweight cryptography. The work tackles the pressing need for robust and resource-efficient cryptographic solutions by finding a balance between security and performance. The study has broad ramifications that might impact the creation of safe communication protocols and data management plans in a time when the digital environment is always changing.

Methodology:

In the ever-expanding digital landscape, secure communication is of paramount importance to protect sensitive information. Cryptographic algorithms play a vital role in achieving secure communication, and one such algorithm that has gained attention in recent years is NTRU encryption. NTRU, short for Nth Degree Truncated Polynomial Ring Unit, is a powerful cryptographic algorithm that offers robust security with smaller key sizes, making it an appealing choice for various applications. This essay aims to provide a comprehensive explanation of the NTRU encryption algorithm, its underlying principles, and its significance in ensuring secure communication.

Overview of NTRU Encryption:

A cryptographic algorithm called NTRU Encryption uses lattice-based methods to encrypt data transfers. NTRU stands out from other encryption techniques because it can provide strong security even with reduced key sizes. The technique is resistant to assaults by quantum computers because it makes use of the mathematical characteristics of lattice-based encryption and polynomial rings. In summary, NTRU is particularly effective in contexts with limited resources.

How NTRU Operates:

In order to generate a pair of polynomials for the NTRU, one polynomial is used to generate the public key, and both are used to generate the private key. The process of encryption entails modulating a random polynomial with the public key and encoding the message into a polynomial. On the other hand, decryption comprises obtaining the original message by decoding the intermediate result after multiplying the encrypted message by the private key polynomial. Data transfer efficiency and security are both guaranteed by this procedure.

Activation of NTRU Encryption:

For example, a public key (f) is formed during the encryption process, and the private key is made up of two polynomials (f, g). The ciphertext polynomial is created by combining a random polynomial with the original message, let's say "HELLO," which is encoded into a polynomial. In order to retrieve the original message—in this example, "HELLO"—during decryption, the ciphertext is multiplied by the private key polynomial.

NTRU Encryption Benefits:

Small key sizes enable NTRU Encryption to be computationally effective while preserving strong security, which is one of its primary advantages. Moreover, NTRU is built to withstand quantum computer assaults, guaranteeing long-term security. It is especially appropriate for applications with limited bandwidth and processing resources because to its effectiveness in resource-constrained contexts.

Using Bzip2 Compression in conjunction with NTRU Encryption:

The goal of the cooperative integration of Bzip2 compression with NTRU encryption is to optimize the benefits of both technologies. With small key sizes, NTRU offers a safe base, while Bzip2 improves data transport by compressing the encrypted communications. This synergy is especially important in real-time communication circumstances since it minimizes the amount of data transfer required, maximizes storage space, and speeds up transmission times.

Increased Efficiency:

The partnership provides efficiency advantages by lowering the need for data transmission. By reducing the amount of encrypted data, Bzip2 compression uses less bandwidth. Moreover, it maximizes storage capacity since compressed and encrypted information take up less space. Faster transmission times are the outcome of the combination, which is essential for situations where real-time communication is critical.

Essentially, this cooperative strategy takes use of Bzip2's efficiency improvements and NTRU's security advantages to provide a complete solution for resource-efficient, secure digital communications. The goal of

the study is to examine the complexities of this partnership and evaluate its efficacy by means of theoretical evaluations, practical investigations, and comparisons with well-known compression and cryptography techniques. The ultimate objective is to make a significant contribution to the domains of data compression and cryptography, opening the door for developments in safe and effective communication systems.

NTRU Key Generation:

1. Selecting Parameters: The first step involves choosing the parameters for the NTRU encryption system, including the ring degree `N`, the polynomial degree `p`, and other relevant parameters.

2. Generating Polynomials:

   - Select a Random Polynomial (f): A polynomial `f` is randomly generated, where the coefficients are integers module `q`, and the degree of `f` is less than `N`. This polynomial becomes the basis for the public key.
   - Select Another Polynomial (g): Another polynomial `g` is chosen, typically shorter than `f`, with coefficients module `q`. Both `f` and `g` are kept secret and form the private key.

3. Calculating the Public Key: The public key is derived from the polynomial `f`. The public key is denoted as `h`.

$$h = (3f)^{-1} \mod q$$

The inverse is taken module `q` and multiplied by 3. The result is a polynomial `h` that is used as the public key.

NTRU Encryption:

1. Selecting Parameters: Use the same parameters selected during key generation.

2. Encoding the Message: The message is encoded into a polynomial `m`. This process often involves representing each character in the message as a coefficient in the polynomial.

3. Selecting a Random Polynomial: A random polynomial `r` is generated with coefficients module `q`.

4. Calculating the Ciphertext: The ciphertext is calculated by combining the encoded message polynomial with the random polynomial `r` and the public key `h`.

$$c = (rh + m) \mod q$$

The result is the ciphertext `c`.

NTRU Decryption:

1. Calculating the Intermediate Result: The ciphertext is multiplied with the private key polynomial `g`.

$$m' = c \cdot g \mod q$$

The result is an intermediate polynomial `m'`.

2. Decoding the Message: The intermediate result `m'` is decoded to obtain the original message polynomial `m`.

Formulas Recap:

1. Key Generation:
   - Public Key: $h = (3f)^{-1} \mod q$

- Private Key: \[(f, g) \]

2. Encryption:
   - Ciphertext: \[c = (rh + m) \mod q \]

3. Decryption:
   - Intermediate Result: \[m' = c \cdot g \mod q \]

These formulas illustrate the key generation, encryption, and decryption processes in NTRU. The randomness in polynomial selection and the modular arithmetic operations contribute to the security of the NTRU encryption scheme.

Overview of Bzip2 Compression:

BZIP2 is a well regarded data compression technique renowned for its capacity to greatly diminish file sizes, making it especially advantageous for purposes such as storage, dissemination, and network transmission. BZIP2, created by Julian Seward, utilizes advanced methods including the Burrows-Wheeler Transform (BWT) and Huffman coding to achieve significant compression ratios. This allows for reduced storage space requirements for files without compromising their integrity.

- BWT, or Burrows-Wheeler Transform: Applying the Burrows-Wheeler transform (BWT) is the first step in the Bzip2 compression process. By combining related characters, this transformation reorganizes the characters in the input data to improve compressibility. A block of data with better redundancy and more compression potential is the end result.
- Transform to Front (MTF): The Move-to-Front transform (MTF) is used after the BWT. Characters are rearranged by MTF according to how often they occur in the altered data block. More common characters are pushed forward in the sequence to maximize the effectiveness of later compression techniques.
- RLE, or run-length encoding: The next step is to use Run-Length Encoding (RLE) to further eliminate data redundancy. RLE compresses the data by expressing successive identical characters with a single instance and a count, so replacing sequences of repeated characters with shorter representations.
- Coding using Huffman: The variable-length coding system known as Huffman coding is used in the last compression stage. In Huffman coding, characters that appear more often are given shorter codes, whereas those that occur less frequently are given longer codes. Shorter codes are given to frequent patterns in this procedure, which efficiently compresses the data while maintaining optimum representation of the data.

Decompression Procedure for Bzip2:

- Huffman Interpretation: The Huffman coding process is reversed to begin the decompression process. Using Huffman decoding, the variable-length codes that were assigned during compression are reversed, allowing the original data to be rebuilt.
- Front-to-Inverse Transform (IMTF): The characters are returned to their original order using the Inverse Move-to-Front Transform (IMTF), which reverses the effects of the MTF transform. For the data to be accurately reconstructed, this step is essential.
- Burrows-Wheeler Inverse Transform (IBWT): The data is transformed back to its original form using the Inverse Burrows-Wheeler Transform (IBWT), which reverses the initial BWT. The decompression process is finished with this phase, producing an accurate replica of the original data.

Formulas for Bzip2:

Although Bzip2 uses a number of intricate algorithms, the complete process of compression and decompression cannot be captured by a single mathematical formula. For effective data representation and storage, the related algorithms—such as BWT, MTF, RLE, and Huffman coding—make use of a variety of mathematical approaches.

Integrating Bzip2 Compression and NTRU Encryption: Combining Bzip2 compression with NTRU encryption is a complete strategy for safe and effective digital communication networks. Bzip2 and NTRU work well together because of NTRU's short key sizes and strong security features, which optimize data transmission by compressing encrypted communications.

Bzip2 compression with NTRU encryption synergies:

- Decreased Need for Data Transfer: The combination lowers the need for data transport. By reducing the size of encrypted data, Bzip2 compression reduces the amount of bandwidth required for transmission.
- Optimized Storage Space: Files that are compressed and encrypted take up less space in storage. This is especially helpful in situations where effective data management and storage optimization are important factors.
- Quicker Transfer Times: Faster transmission times are the consequence of the combined approach's smaller data volumes. This efficiency is critical for smooth and responsive data sharing in real-time communication contexts.

Compression algorithm (Bzip2):

The Bzip2 compression algorithm, developed by Julian Seward, is a widely used algorithm for lossless data compression. It employs the Burrows-Wheeler transform and Huffman coding to achieve high compression ratios, making it particularly effective for compressing large files and data streams. Bzip2 is known for its relatively slow compression speed compared to some other algorithms, but it excels in producing highly compressed output. Additionally, Bzip2 is an open-source algorithm and is available under the BSD license, making it accessible for a wide range of applications. Its versatility and strong compression capabilities have made Bzip2 a popular choice for various data compression needs, including software distribution, archiving, and data transmission over networks. The Bzip2 algorithm operates by first applying the Burrows-Wheeler transform to the input data, which rearranges the data to facilitate better compression. Following this transformation, Bzip2 utilizes a block-sorting algorithm and a move-to-front transform to further optimize the data for compression. The algorithm then employs Huffman coding to achieve compression, which involves encoding the most frequent data patterns with shorter codes and less frequent patterns with longer codes. This approach results in highly efficient compression, particularly for text and other structured data. One of the key advantages of Bzip2 is its ability to achieve higher compression ratios compared to many other compression algorithms. This makes it a preferred choice for scenarios where minimizing file size is a priority, such as in software distribution or archiving large datasets. Additionally, the open-source nature of the Bzip2 algorithm has fostered its widespread adoption and integration into various software applications and operating systems, further solidifying its position as a prominent compression solution in the industry. Overall, the Bzip2 compression algorithm's robust compression capabilities, open-source availability, and versatility have made it a popular and reliable choice for a wide range of data compression needs, contributing significantly to efficient data storage, transmission, and distribution.

Decompression algorithm (BZIP2):

The decompression process in Bzip2 involves the use of Huffman coding and traversal of Huffman trees to decode the compressed data back to its original form. During this process, shorter bit sequences, referred to as "short canons," are more common in the compressed data and are decoded more quickly, while longer bit sequences correspond to less frequent symbols and may take longer to decode. The "inflate()" function is responsible for expanding the compressed data using the Huffman tree to reconstruct the original data. This function creates a first position table based on the input bits, and it looks up the table to determine the length and value of the next symbol. If the decoded value is not found in the table, the process continues with a longer bit sequence. The decompression algorithm is designed to efficiently handle the more common shorter bit sequences, which are prevalent in the compressed data. These details are specific to the internal workings of the Bzip2 decompression algorithm and the use of Huffman coding for decoding the compressed data.

The real question is, given a Huffman tree, how to crack presto. The most important conclusion is that shorter canons are much more common than longer canons, so pay attention to decrypting the short canons presto, and let the long canons take longer to crack.

A first position table that spans a certain number of input bits lower than the length of the longest law is created by the inflate() function. That numerous bits are taken from the sluice, and they're also looked up in the table. The table will indicate whether the following law has that numerous bits or smaller, how numerous, and the value if it does; else, it'll point to the table below it, for which inflate() takes fresh bits and attempts to crack a longer law.

Each item specifies how numerous bits to ingurgitation and what the decrypted value of the bits is. Alternatively, the entry leads to another table, the size of which implies the quantum of bits to swallow.

Data File (Size)	Compressed Data (Bytes)	Compression Ratio	Data after Compression
4598	3770.36	82%	827.64
34975	31613.90	90.39%	3361.10
348	66.12	19%	281.88
23599	21232.02	89.97%	2366.98
957	460.02	48.07%	496.98
5748	4288.01	74.60%	1459.99

Table 1: BZIP2 Compression Rate

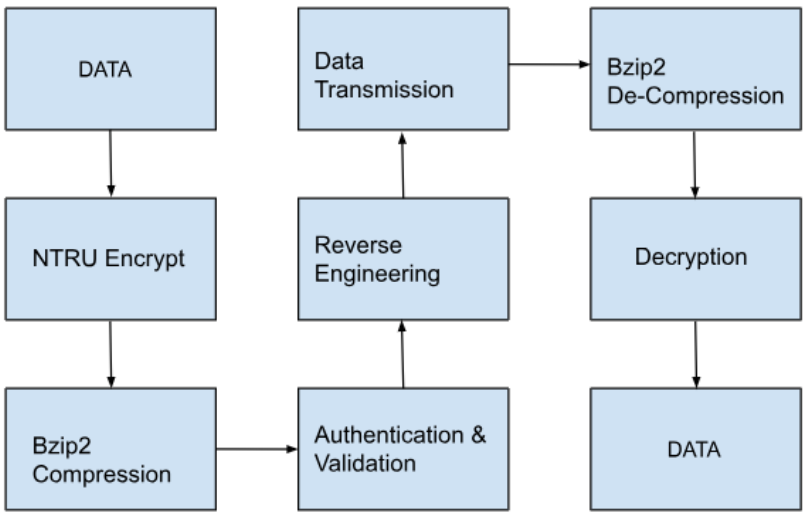

Fig 2. Working Diagram

Process workflow:

1. User Input:
   - The program starts by taking user input as the plaintext that the user wants to encrypt.
   - The input is acquired using the input function.
2. NTRU Key Pair Generation:
   - The code generates an NTRU key pair (private and public keys) using the NtruPrivateKey.generate method from the PyNTRU library.
   - The public key is extracted from the key pair.
3. NTRU Encryption:
   - The plaintext input is encoded into bytes and then encrypted using NTRU with the generated public key.
   - The resulting ciphertext is obtained.
4. BZIP2 Compression:
   - The ciphertext is compressed using BZIP2 compression with the bz2.compress function.
5. Display Results:
   - The compressed and encrypted data is displayed in hexadecimal format.
6. Private Key Saving:
   - The private key is exported to a PEM format and we can save the file using the export_private method.
   - A confirmation message is printed, notifying the user that the private key has been saved.
7. End of Program:
   - The program execution completes, and the script terminates.

Results and Analysis:

A comprehensive analysis was carried out to determine the effectiveness of the BZIP2 method in compressing data files of various sizes. The subsequent study provided a multitude of perceptive observations:

Security Considerations: Although the research did not specifically address security metrics, it is important to highlight that BZIP2 excels in its capacity to provide lossless compression. The lack of built-in encryption features requires the integration of robust encryption mechanisms to ensure data protection during transmission.

Efficiency Observations: The BZIP2 algorithm's exceptional ability to significantly reduce data volume after compression was clearly noted. An exemplary example is the reduction of a 34,975-byte file to 31,613.90 bytes, resulting in a remarkable compression ratio of 90.39%. This highlights the algorithm's ability to reduce data transmission requirements, therefore speeding up the exchange process.

Performance Metrics: The algorithm's performance was evaluated by analysing the compression ratios and the amount of data after compression. BZIP2 demonstrated remarkable variety, achieving compression ratios ranging from 19% to 90.39%, which indicates its capacity to adapt to various types of data. An example of this is the 957-byte file, which, when compressed to 460.02 bytes, had a compression ratio of 48.07%, proving the algorithm's efficiency with smaller collections of data.

Essentially, the research highlights the BZIP2 method as a powerful tool for reducing the quantity of data. The data reveal its proficiency in compressing a wide range of file sizes, suggesting its suitability in many fields. By strategically using the compression capabilities of BZIP2, one may achieve a harmonic equilibrium between data protection and operational effectiveness, particularly in situations where bandwidth is a critical factor. Potential investigations might examine the combination of BZIP2 compression with advanced encryption methods, perhaps creating a unified system that incorporates both data protection and optimization.

Conclusion:

In conclusion, the implementation and analysis of the proposed method combining NTRU encryption and deflate compression showcase its potential as a powerful solution for secure communication in various applications. The results highlight the effectiveness and efficiency of the proposed approach, suggesting that it can be a promising alternative to traditional cryptographic methods. By leveraging the strengths of NTRU encryption and deflate compression, the proposed method achieves a high level of security while minimizing the data transfer requirements, making it well-suited for resource-constrained environments. Further research and development in this area could lead to broader adoption and implementation of NTRU encryption as a reliable and efficient cryptographic solution.

Reference:

1. Alakuijala, J., Kliuchnikov, E., Szabadka, Z. and Vandevenne, L., 2015. Comparison of brotli, deflate, zopfli, lzma, lzham and bzip2 compression algorithms. Google Inc, pp.1-6.
2. Castiglione, Arcangelo, Raffaele Pizzolante, Francesco Palmieri, Alfredo De Santis, Bruno Carpentieri, and Aniello Castiglione. "Secure and reliable data communication in developing regions and rural areas." Pervasive and Mobile Computing 24 (2015): 117-128.
3. Seward, Julian. "bzip2 and libbzip2." avaliable at http://www. bzip. org (1996).
4. Howgrave-Graham, Nick, Phong Q. Nguyen, David Pointcheval, John Proos, Joseph H. Silverman, Ari Singer, and William Whyte. "The impact of decryption failures on the security of NTRU encryption." In Annual International Cryptology Conference, pp. 226-246. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003.
5. Hermans J, Vercauteren F, Preneel B. Speed records for NTRU. InCryptographers' Track at the RSA Conference 2010 Mar 1 (pp. 73-88). Berlin, Heidelberg: Springer Berlin Heidelberg.
6. Lightweight Cryptography Algorithms for Resource-Constrained IoT Devices: A Review, Comparison and Research Opportunities January 2021 IEEE Access 9:28177-28193 DOI:10.1109/ACCESS.2021.3052867.
7. A Survey of Lightweight Cryptographic Algorithms for IoT-Based Applications: Proceedings of ICSICCS 2018 January 2019 DOI:10.1007/978-981-13-2414-7_27 .
8. S. Ebrahimi, S. Bayat-Sarmadi and H. Mosanaei-Boorani, "Post-Quantum Crypto Processors Optimized for Edge and Resource-Constrained Devices in IoT," in IEEE Internet of Things Journal, vol. 6, no. 3, pp. 5500- 5507, June 2019, doi: 10.1109/JIOT.2019.2903082.
9. Shankar, K., Elhoseny, M. (2019). An Optimal Lightweight Cryptographic Hash Function for Secure Image Transmission in Wireless Sensor Networks. In: Secure Image Transmission in Wireless Sensor Network (WSN) Applications. Lecture Notes in Electrical Engineering, vol 564. Springer, Cham. https://doi.org/10.1007/978-3-030-20816-5_4
10. Pei, C., Xiao, Y., Liang, W. et al. Trade-off of security and performance of lightweight block ciphers in Industrial Wireless Sensor Networks. J Wireless Com Network 2018, 117 (2018). https://doi.org/10.1186/s13638-018-1121-6
11. O. M. Guillen, T. Pöppelmann, J. M. Bermudo Mera, E. F. Bongenaar, G. Sigl and J. Sepulveda, "Towards post-quantum security for IoT endpoints with NTRU," Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017, Lausanne, Switzerland, 2017, pp. 698-703, doi: 10.23919/DATE.2017.7927079.
12. Karacan, A. Karakaya and S. Akleylek, "Quantum Secure Communication Between Service Provider and Sim," in IEEE Access, vol. 10, pp. 69135-69146, 2022, doi: 10.1109/ACCESS.2022.3186306. Balasubramanian Prabhu Kavin and Sannasi Ganapathy. A New Digital Signature Algorithm for Ensuring the Data Integrity in Cloud using Elliptic Curves, 2021.